Instituto Tecnológico de Costa Rica

Segundo Avance: Definición Intérprete

Integrantes:

Josué Arce González

Daniel Montero Carvajal

Curso de Compiladores e Interpretes

01/06/2018

Contenido

[Introducción 3](#_Toc515570418)

[Soluciones e Implementación 3](#_Toc515570419)

[Resultados Obtenidos 3](#_Toc515570420)

[Manual de Pruebas 3](#_Toc515570421)

[Conclusiones 3](#_Toc515570422)

[Bibliografía 3](#_Toc515570423)

# Introducción

Interpretar instrucciones es un proceso arduo, elaborado, el cual requiere de mucho análisis, ya que se deben de tener en cuenta todos los posibles casos que se puedan dar, evaluar la sintaxis escrita, verificar si cumple con lo solicitado y mostrarle un resultado al respecto al usuario; de esto trata el siguiente documento, hace énfasis a las pruebas hechas sobre el intérprete generado, se muestran los resultados obtenidos, secciones que no se cumplieron y porqué, además, se adjuntan las soluciones tomadas y su respectiva explicación, por último, un pequeño comentario que refleja nuestras opiniones luego de haber culminado con el susodicho avance.

# Soluciones e Implementación

A continuación, se detallan los elementos utilizados que fueron la solución a la etapa #3:

1. EvaluationStack:
   1. Encargado de almacenar durante un período todos los resultados, valores relacionados a una variable, función, entre otros, para así manipular de los mismos y generar una respuesta, se dice que contendrá los datos de manera periódica, porque estos van a estar constantemente en cambio, es decir, se introduce un valor, utilizando el método ***pushValue***, el cual recibe un valor de cualquier tipo y se inserta en la pila,pero este luego será obtenido para aplicarle a este un cálculo matemático, y luego se inserta el nuevo valor de nuevo a la pila; por otro lado, también existe un método llamado ***popValue*** el cual permite obtener el valor que se encuentre en el “tope” de la pila.
2. Interpreter:
   1. Encargado de generar todos los resultados a partir de una, dos o más variables, métodos, entre otros, dependiendo del cálculo solicitado este va realizar el proceso necesario para así poder generar una respuesta (error o valor) a partir de lo solicitado; cada método creado en esta sección tiene un propósito importante, hay métodos que están enlazados, es decir, uno depende del resultado de otro, por lo que fue importante al inicio pensar en una estrategia para que no hubiesen atrasados o errores conforme se desarrollaba el programa o a futuro.
   2. Conforme se van visitando los métodos necesarios para fabricar el resultado, se van realizando al mismo las validaciones necesarias para que el proceso no llegue a fallar, se debe siempre mostrar un resultado al usuario; por cada método se obtienen el primer, los 2 primeros, o más valores que se encuentren en el “tope” de la pila, claro está que al inicio de programa se debe establecer una instancia de la clase ***evaluationStack***¸ para así poder acceder a los métodos dentro de esta, entonces conforme se vayan necesitando de valores se extraen de la pila, se les aplica una fórmula matemática y el paso siguiente es insertar el resultado de dicho calculo en la misma pila, ya que se va necesitar luego tanto para otro calculo o para mostrarlo en pantalla.
3. DataStorage:
   1. Es la base de esta tercera etapa, ya que toda la información ingresada en el IDE, se va almacenar en esta sección, absolutamente todo se almacena, con el siguiente orden: **nombre de la variable -> valor asignado en el IDE**; esto permite poder tener el identificador junto a su respectivo valor, entonces su principal uso es el poder acceder a un identificador en específico, para esto se elaboraron dos métodos, los cuales son llamados ***getData***, uno recibe la posición exacta de lista donde están concentrados los datos, y el segundo método recibe el nombre del identificador, para así buscarlo a partir del respectivo nombre asignado; otro método importante es el ***cleanData***, este nos permite eliminar todos los registros que se encuentren actualmente dentro de la lista, entonces en pocas palabras lo que realiza es vaciar completamente la lista. Pero no podemos dejar de lado el método más importante de esta sección, y es la función llamada ***addData***, este método recibe 3 parámetros, el nombre asignado, el valor establecido para dicho identificador y la posición actual de la lista, entonces se introduce el nuevo registro en la posición recibida + 1, esto para asignarla en la siguiente posición de la lista enlazada.

# Resultados Obtenidos

|  |  |  |
| --- | --- | --- |
| Aspecto Por Evaluar | Estado (0%-100%) | Justificación en caso de no estar el aspecto al 100% |
| DataStorage | Se encuentra funcionando en perfectas condiciones, introduce y almacena cualquier tipo de dato sin problemas. | N/A |
| EvaluationStack | Se encuentra funcionando en perfectas condiciones, almacena y permite introducir y acceder a todos los valores introducidos. | N/A |
| Interprete | Se encuentra funcionando en perfectas condiciones, todas las casos posibles fueron cubiertos, cualquier tipo de función matemática, consulta deseada retorna un valor. | N/A |
| Estado General | El avance # 3 fue abarcado correctamente dentro del ámbito de las pruebas realizadas. | N/A |

# Manual de Pruebas

# Conclusiones

Exitosamente se lograron abarcar los puntos establecidos al inicio, el programa realiza lo solicitado por el usuario y siempre muestra un resultado al respecto, el intérprete funciona a la perfección lo cual genera total satisfacción ya que se puede ver que el proceso invertido en funcionamiento; luego de haber experimentado toda la creación de un sistema compilador, se puede responder con seguridad, que no es nada fácil, es para nada sencillo generar un programa que permita leer un código y mostrar un resultado, es un largo y complejo proceso, el cual debe manejarse con sumo cuidado y abarcando todas las etapas, cabe destacar que en el desarrollo del mismo solo se cubrieron 3 etapas, pero faltaron muchas más, por ejemplo, el generar un archivo con el código fuente, además, de que el lenguaje ***Java*** te asiste de gran manera, te brinda muchas opciones al momento de programar, por lo que resulta de una u otra manera un peso menos.

Es increíble ver todos esos compiladores ya creados, que están en funcionamiento, y ahora se puede concluir el respeto y sobre todo valorar el trabajo dedicado por los desarrolladores de código, ya que gracias a ellos tenemos la dicha de poder escribir programas. El intérprete para terminar es una base fundamental a tener en cuenta, dentro de este ocurre la magia, es decir, es aquí donde las operaciones aritméticas, comparaciones, llamadas a métodos se deben de evaluar a la perfección, el susodicho genera la respuesta final para el usuario.

# Bibliografía